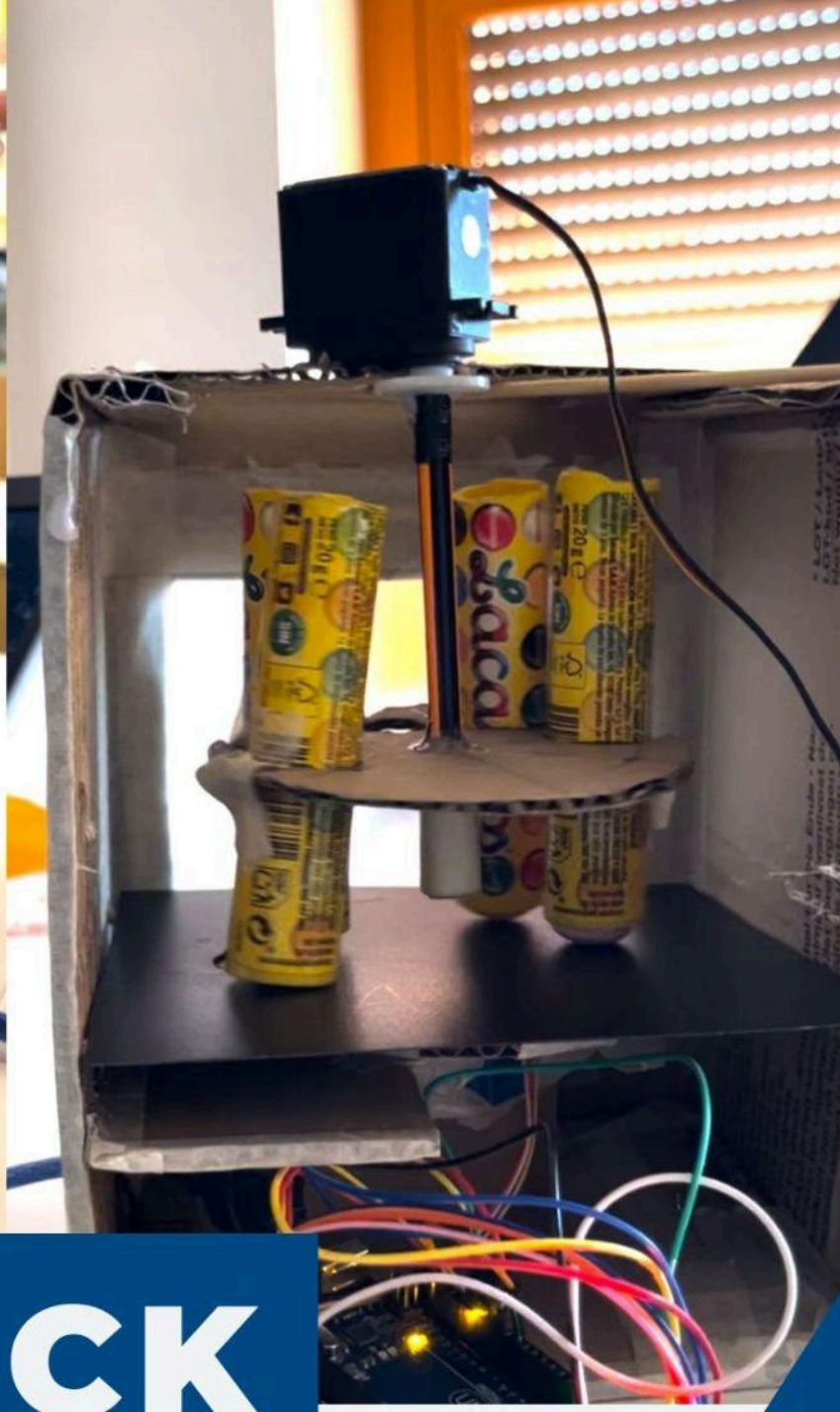


MEDILOCK

SISTEMA AUTOMATIZADO DE DISPENSACIÓN DE MEDICAMENTOS

CLAUDIA GALLEGO MOLINA
ROCÍO MIGUEL MUÑOZ
LUCIA SEGRELLES MONTERO
LUCAS RODRIGUEZ DIEZ

ÍNDICE

1. INTRODUCCIÓN **2**
2. OBJETIVOS **2**
3. SOFTWARE **3**
4. HARDWARE **4**
5. FUNCIONAMIENTO MECÁNICO **5**
6. COMUNICACIÓN EN SERIE **7**
7. PRESUPUESTO **7**
8. PROBLEMAS Y SOLUCIONES **7**
9. ANEXO **10**

1. INTRODUCCIÓN

En muchos centros hospitalarios o residencias, la gestión de los medicamentos ha sido un importante quebradero de cabeza para muchos profesionales. La alta cantidad de pacientes y la gran cantidad de medicamentos existentes hoy en día provoca serias dificultades a la hora de suministrar de forma correcta los medicamentos a cada paciente. Ya que una simple equivocación puede tener consecuencias muy graves para la salud de enfermos.

De hecho, hoy en día, existen denuncias interpuestas contra residencias debido a errores de medicación. Ejemplo de ello es una residencia de mayores en Alcorcón, gestionada por la Consejería de Familia y Asuntos Sociales. El fallo judicial da por probado el testimonio de un médico de la residencia que alertaba de irregularidades en el suministro de medicamentos.

El personal sanitario asume una gran carga de trabajo a lo largo del día preparando y entregando medicación, lo cual puede desembocar en una confusión en la entrega de medicinas. Debido a ello y, con la idea de mejorar tanto la seguridad del paciente como la eficacia del personal que trabaja en hospitales y residencias, hemos decidido desarrollar un proyecto que solucione todas estas problemáticas.

2. OBJETIVOS

Los objetivos técnicos que se desean cumplir son los siguientes:

- Desarrollo de una aplicación en Java que permita gestionar medicamentos y pacientes de forma intuitiva.
- Desarrollo de un hardware controlado desde Arduino que permita la dispensación de medicamentos.
- Establecer una comunicación correcta y fiable entre los componentes anteriormente descritos.

Los objetivos funcionales que se desean cumplir son los siguientes:

- Permitir a los sanitarios acceder con sus credenciales
- Manejar una lista de los pacientes con cada una de sus pastillas
- Manejar un stock individualizado a cada paciente
- Permitir la dispensación de forma automática únicamente con un click.

3. SOFTWARE

El funcionamiento de nuestro proyecto se dirige desde una aplicación creada en java que se encarga tanto de manejar la autenticación del personal como de la dispensación de los propios medicamentos.

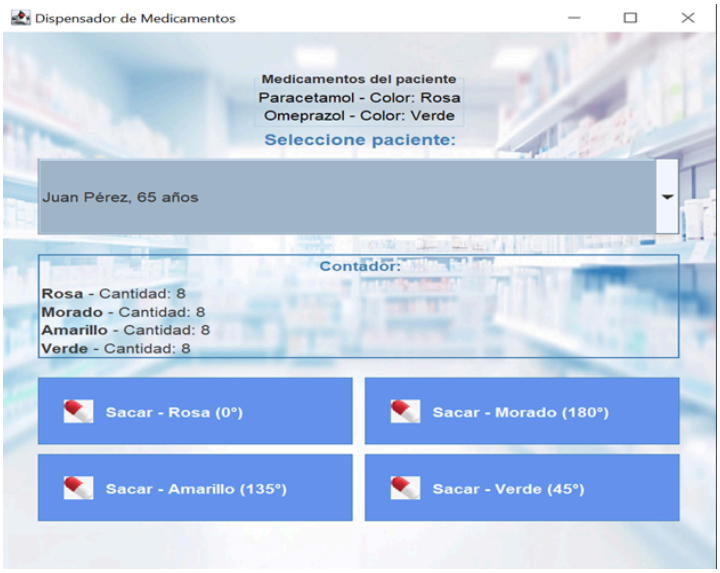
A continuación, se muestra una vista de la primera de las ventanas, que desarrolla la funcionalidad de log in. Únicamente podrán acceder a la aplicación aquellas personas que aparezcan registradas. Las personas que proponemos que estén registradas son los enfermeros y médicos pertenecientes a centros hospitalarios o residencias. El usuario es el nombre y apellido del usuario, con ambas iniciales en mayúsculas. (Ej: PedroPerez). Se recomienda poner una contraseña fiable y segura que permita únicamente entrar al propio usuario para evitar posibles usos fraudulentos de nuestro software. Es importante destacar que si una persona no está registrada o erra en su contraseña/usuario, salta una notificación advirtiendo del fallo.

La segunda ventana diseñada maneja la dispensación de medicamentos, en ella podemos distinguir distintos componentes:

a) Medicamentos del paciente, situado en la parte superior de la ventana, que muestra los medicamentos asociados a cada paciente, imposibilitando la dispensación de un medicamento que tenga en tratamiento el paciente.

b) Listado de pacientes que permite seleccionar los pacientes asociados a un centro o residencia.

c) Un stock que se va actualizando a medida que se dispensan los medicamentos. El stock es diario y



cada día se renueva con las pastillas que necesite el paciente, permitiendo además saber cuántas pastillas le quedan a cada paciente.

d) Cuatro botones asociados a las pastillas, estos botones al ser pulsados mandan un ángulo distinto a la placa de arduino para que se muevan los motores y dispensen cada una de las pastillas. Se encuentran impresos grados que se tienen que mover los motores para poder dispensar las pastillas.

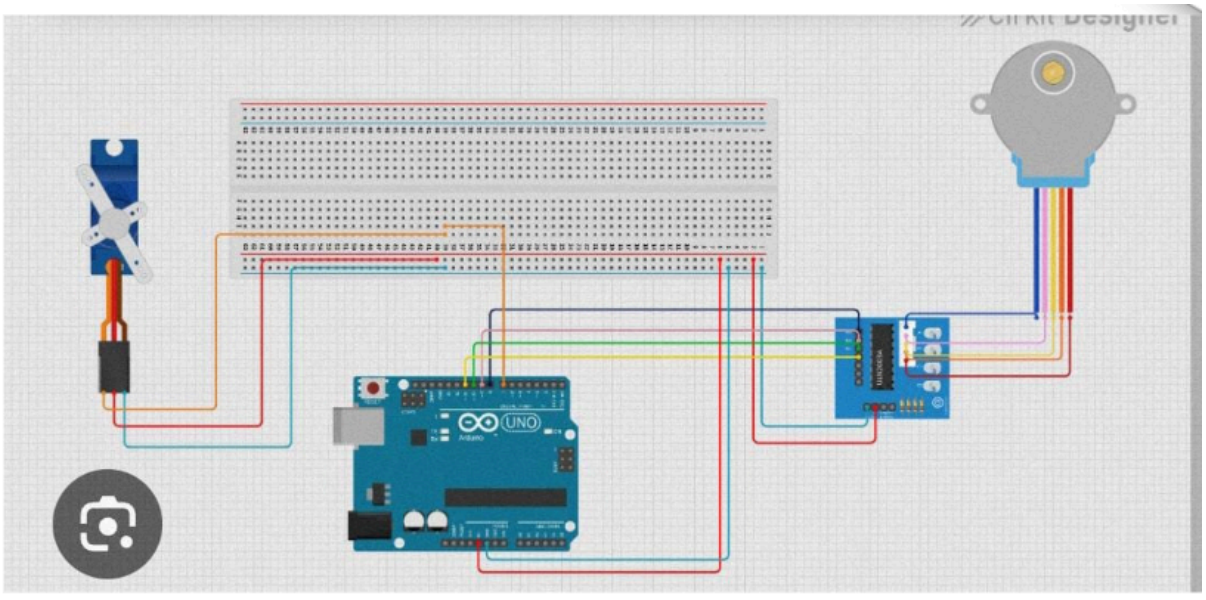
4. HARDWARE

Los componentes utilizados han sido los siguientes.
Para el funcionamiento:

- 3 cables M-M
- 6 cables M-H
- Placa Arduino UNO
- Cable USB
- 1 Servomotor
- 1 Stepmotor 28BYJ-48
- Driver ULN2003
- Código Java e Interfaz Gráfica

Funcionamiento Motores:

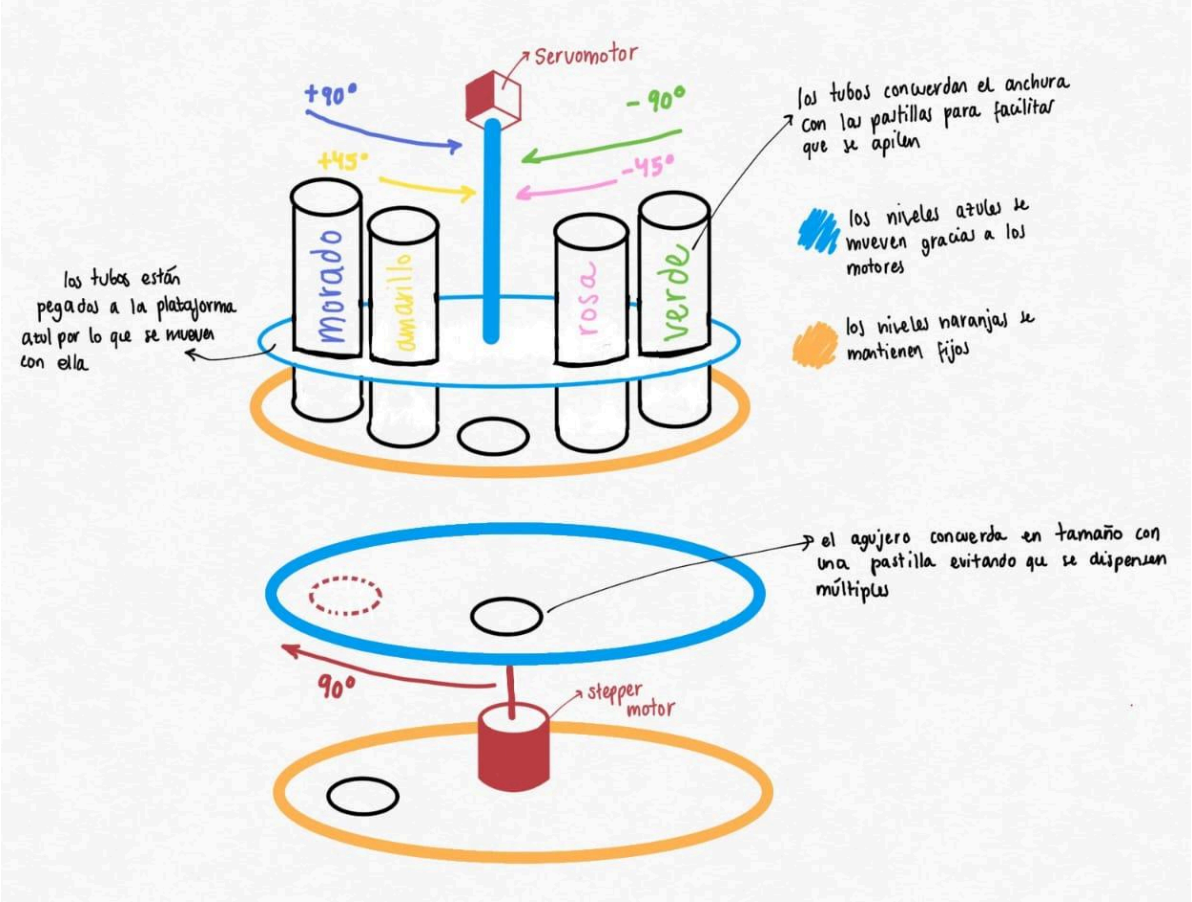
A. Servomotor: conectado a tierra (GND), a los 3.3V y al pin digital 6
B. StepMotor: los pines de control van conectados a los pines digitales 8,9,10 y 11, conectado a tierra (GND) y a los 5V

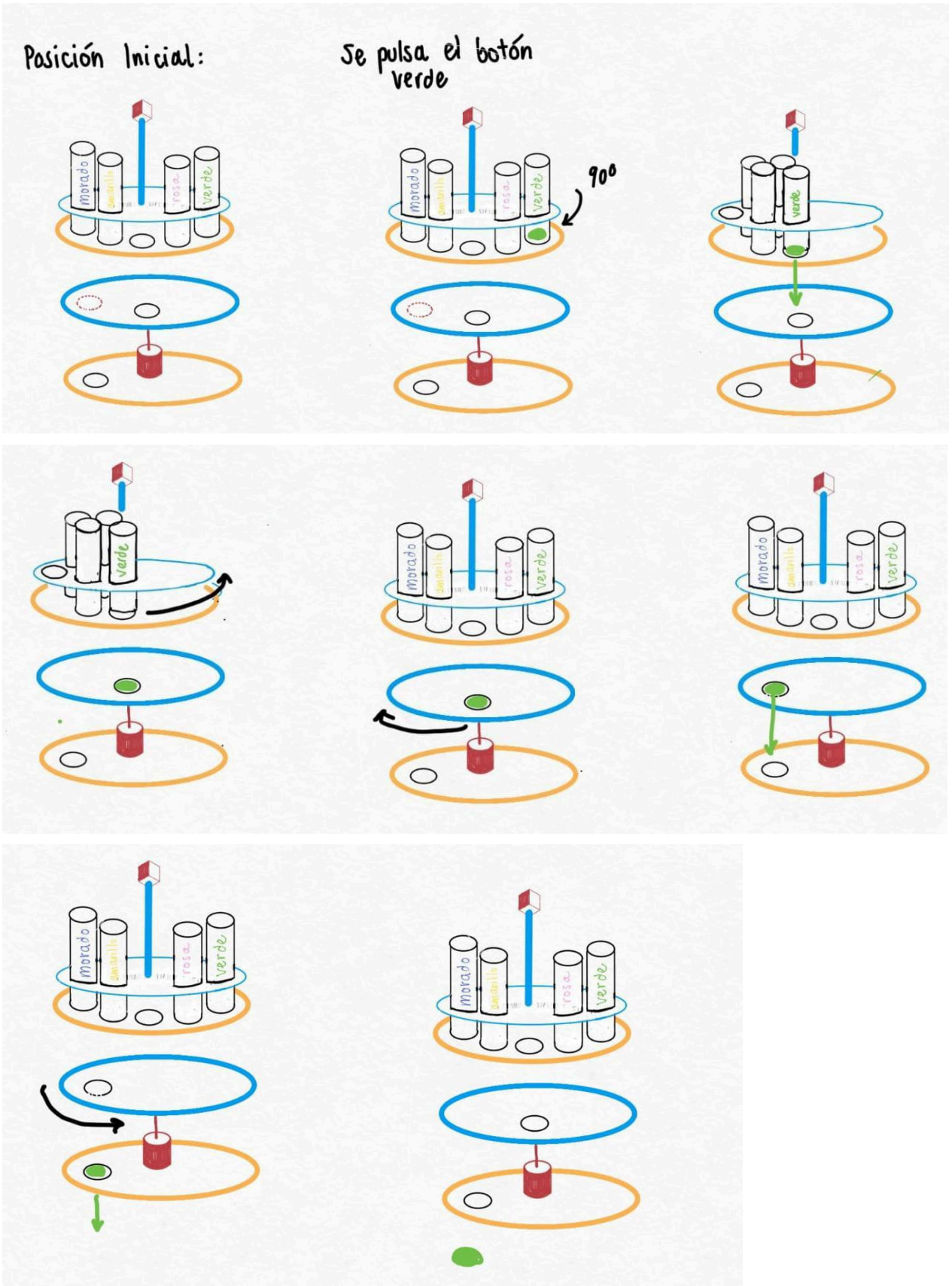


Para el montaje

- Caja de cartón
- Carpetas de plástico
- Pistola de silicona
- Botes lacasitos
- Mentos
- Cuter

5. FUNCIONAMIENTO MECÁNICO

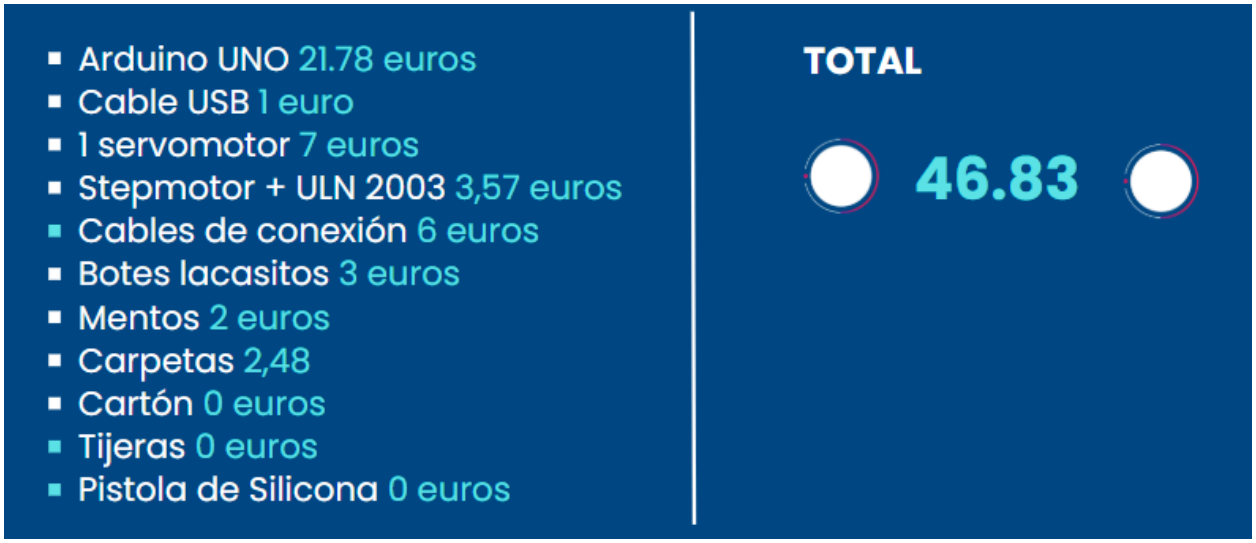



Posición Inicial:
Se pulsa el botón verde
morado
amarillo
rosa
verde
90º

6. COMUNICACIÓN EN SERIE

Se comunica el código Java con Arduino a través del puerto serie, enviando comandos para que el motor gire diferentes ángulos hasta 180º según el medicamento elegido. Este es posible gracias a la librería JSerialComm de Java y las clases ArduinoControl y SerialComunicación implementadas en el código

7. PRESUPUESTO



8. PROBLEMAS Y SOLUCIONES

Para comenzar, explicaremos cuál era la idea principal que teníamos al iniciar este proyecto y cómo ha ido evolucionando hasta llegar a la solución final.

En un principio, nuestra intención era desarrollar una aplicación para Android, una APK, que se conectara con la placa de Arduino utilizando un módulo Bluetooth HC-05. La idea era que, desde el móvil, se pudiera elegir el medicamento y enviar la orden de forma inalámbrica.

Sin embargo, cuando empezamos con las pruebas para esta opción, nos encontramos con problemas de compatibilidad entre la APK que desarrollábamos y el módulo Bluetooth HC-05. A pesar de varios intentos, no conseguimos establecer una comunicación estable ni funcional entre el móvil y el Arduino, por lo que decidimos descartar esta idea.

Después de nuestra primera opción, probamos otra alternativa: usar MIT App Inventor para montar la aplicación. Al principio, aunque parecía una opción más sencilla e intuitiva, a la hora de llevarlo a un móvil real y realizar pruebas, también nos encontramos las mismas dificultades técnicas para que el sistema funcionara correctamente, ya que no conseguíamos una comunicación fiable con el hardware.

Finalmente, tras evaluar todas las opciones presentes, nos decantamos por desarrollar una aplicación sencilla en Java para escritorio que manda datos por serial gracias a la librería de Java por jSerialCom, que se conecta a Arduino a través de un cable USB.

Esta decisión fue clave, porque nos permitió cumplir todas las funcionalidades sin complicaciones técnicas. La conexión por puerto serie fue directa y estable, y al trabajar en un entorno de escritorio, el control sobre el comportamiento de la aplicación y la integración con el hardware es mauor.

Así que, aunque nuestra idea original era algo diferente y más enfocada al móvil, al final optamos por una solución más práctica, que nos permitió centrar los esfuerzos en que el sistema funcione de forma fiable y completa.

Otro de los problemas que encontramos fue que habíamos pensado usar los lacasitos como si fueran las pastillas, y por ello los compramos ya en sus tubos. Sin embargo, no eran tan anchos como pensábamos, lo que impedía que se apilaran correctamente en el bote y dificultara su dispensado. Finalmente, decidimos cambiar a mints, ya que tenían el tamaño adecuado para apilarse bien.

Otro de nuestros inconvenientes fue que el servomotor, colocado en un principio abajo, chocaba con los tubos de lacasitos al intentar girar, por lo que perdía movilidad. Para solucionar esto, elevamos el servomotor con la ayuda de un lápiz para que tuviera más espacio y cubriera completamente todos los ángulos.

Por último, la mayor dificultad ha sido ajustar los ángulos, ya que la fricción entre los materiales hacía que, con el más mínimo movimiento, se descalibrara el sistema, y que no se dispensaran correctamente todas las pastillas. Además, en Arduino es complicado ajustar los ángulos con la precisión necesaria, por lo que hemos ido calibrando el sistema poco a poco hasta conseguir el funcionamiento adecuado.

9. ANEXO

```ino
#include <Servo.h>
#include <Stepper.h>

// Definición del servo motor
Servo servo;
const int pinServo = 6;

// Configuración del motor paso a paso
const int pasosPorVuelta = 2048;
Stepper stepper(pasosPorVuelta, 8, 10, 9, 11); // Pines conectados al ULN2003

void setup() {
  Serial.begin(9600);           // Inicializar comunicación serie
  servo.attach(pinServo);       // Asignar pin al servo
  servo.write(90);              // Posición inicial (reposo)
  stepper.setSpeed(10);         // Velocidad del motor paso a paso (RPM)
}

void loop() {
  if (Serial.available()) {
    String entrada = Serial.readStringUntil('\n'); // Leer hasta salto de línea
    entrada.trim();                                // Eliminar espacios
    int angulo = entrada.toInt();                  // Convertir a número

    if (angulo >= 0 && angulo <= 180) {
      servo.write(angulo);      // Mover el servo al ángulo recibido
      delay(2000);              // Espera de 2 segundos simulando dispensado
      servo.write(90);          // Volver a la posición inicial (90°)

      int pasos90grados = pasosPorVuelta / 4; // 90° en pasos
      stepper.step(pasos90grados);            // Mover la bandeja
    }
  }
}
```